CMPG-767 Image Processing and Analysis

LOCAL NEIGHBORHOOD PROCESSING. CONCEPT OF SPATIAL DOMAIN FILTERING

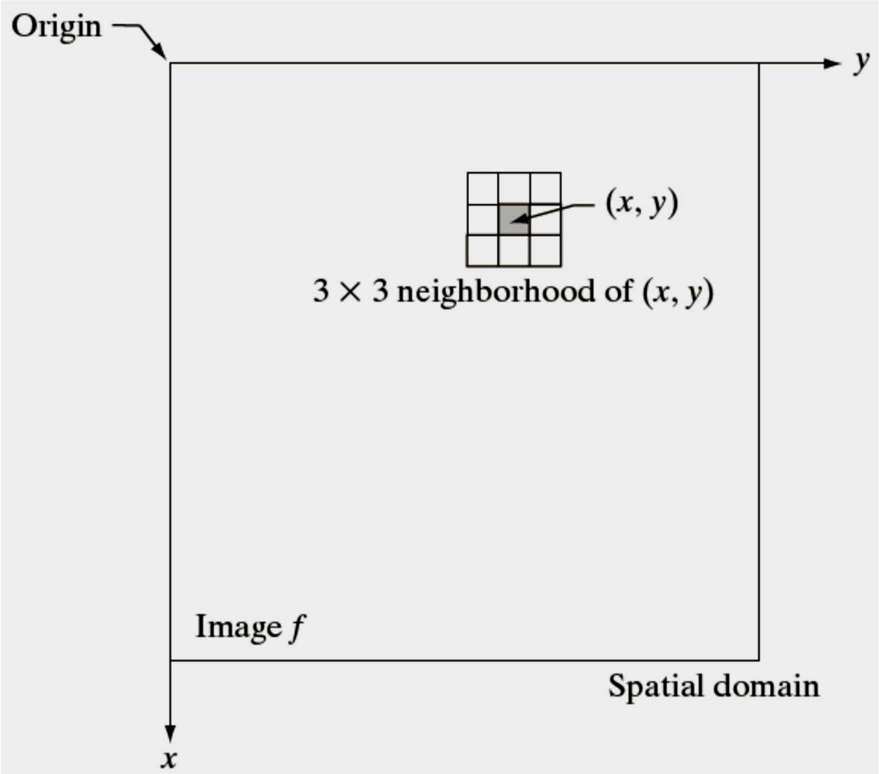
Spatial Domain

- The Spatial Domain is a domain where a digital image is defined by spatial coordinates of its pixels

- (Another domain considered in image processing is the frequency domain where a digital image is defined by breaking down into the spatial frequencies participating in its formation – we will consider it later)

Local Neighborhood in Image Processing

- Many image processing operations in the spatial domain (particularly, spatial domain filtering) are reduced to local neighborhood processing

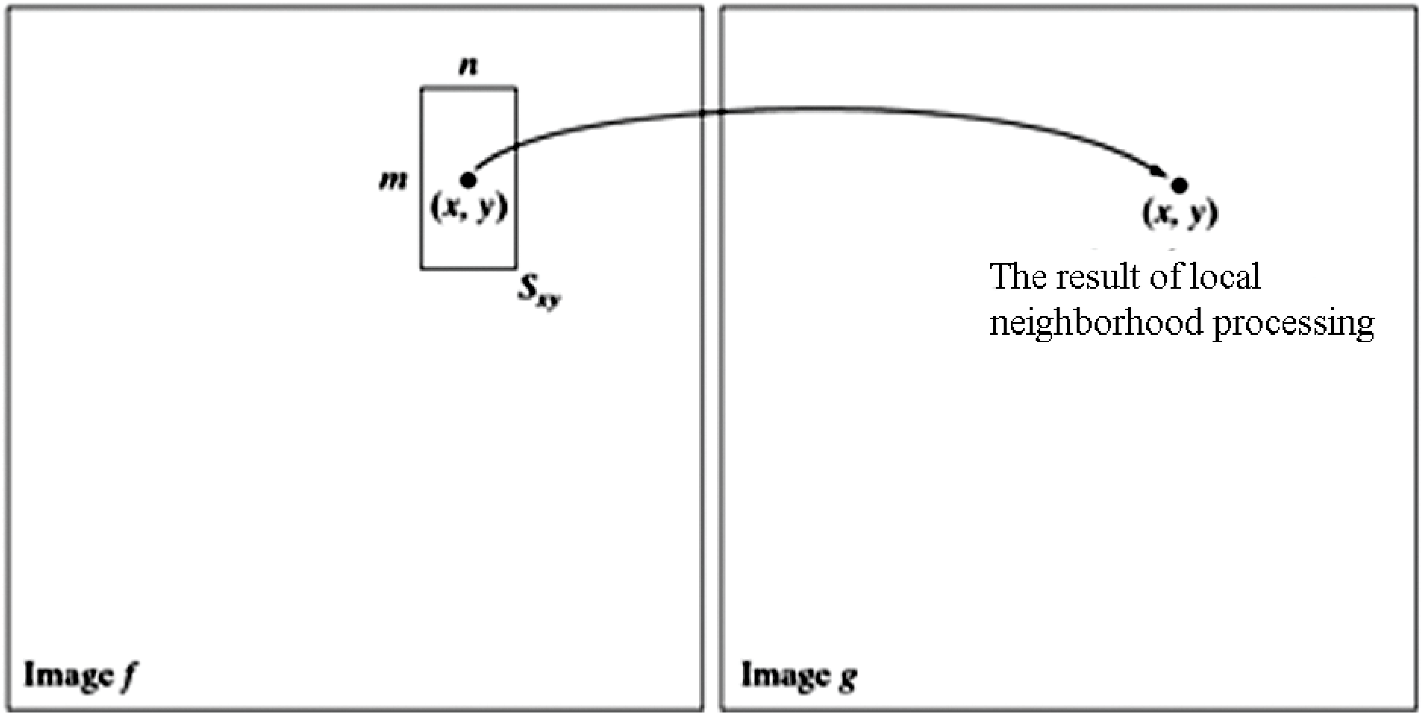


Local Neighborhood in Image Processing

- Let S_{xy} be the set of coordinates of a neighborhood centered on an arbitrary pixel (x, y) in an image f

- Neighborhood processing generates a corresponding pixel at the same coordinates in an output image g, such that the intensity value in that that pixel is determined by a specific operation involving the pixels in the input image with coordinates in S_{xy}

Local Neighborhood in Image Processing



Local Neighborhood in Spatial Domain Image Processing

- The spatial domain processing can be represented by the following expression

$$g(x,y) = T\left(f(x,y)\right)$$

where $f(x,y)$ is an input image, $g(x,y)$ is an output image and T is an operator defined over a local neighborhood of pixel with the coordinates (x, y)

Local Contrast Enhancement

- Local contrast enhancement is utilized through local histogram processing
- The local histogram in an *n* x *m* window S_{xy} of a digital image with intensity levels $\{0,1,\ldots, L\text{-}1\}$ is a discrete function

$$h_{S_{xy}}\left(\begin{matrix} r_k \\ S_{xy} \end{matrix}\right) = n_k ; k = 0,1,\ldots, L-1$$

Local Contrast Enhancement

- Local contrast enhancement leads to better visibility of local details and, as a result, to the extraction of those details (not necessarily small details) that are hidden in the areas with a poor local contrast
- Local contrast enhancement can be utilized, for example, through the same methods that are used for the global contrast enhancement

Local Contrast Enhancement

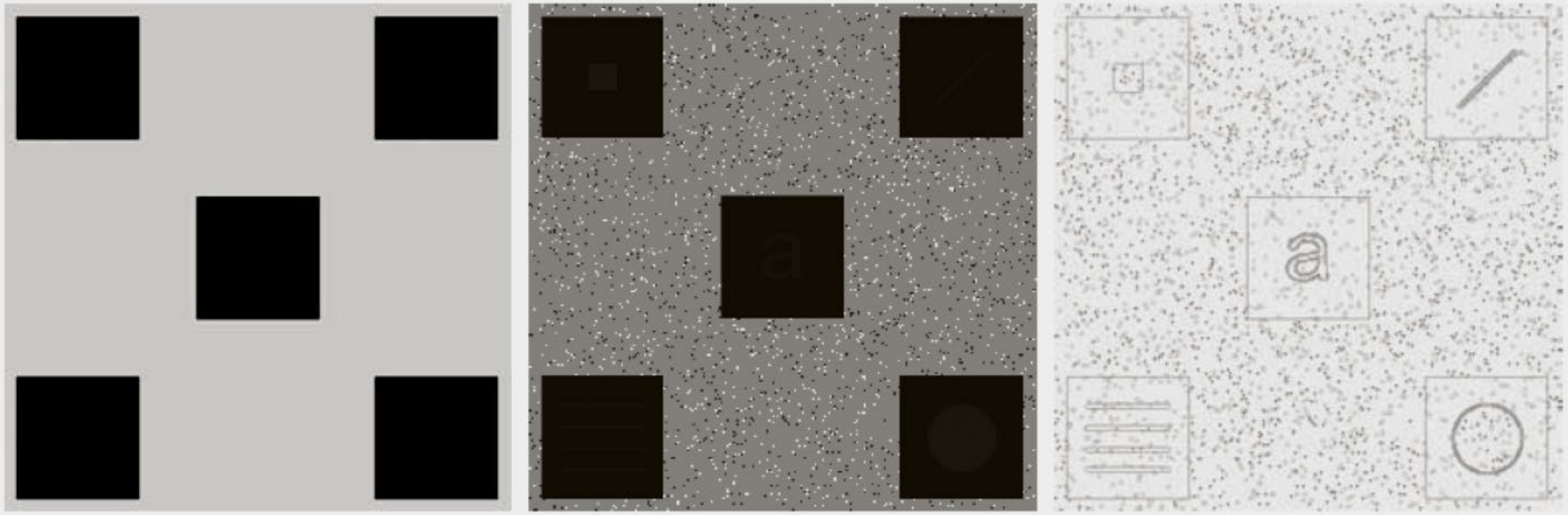


FIGURE 3.26 (a) Original image. (b) Result of global histogram equalization. (c) Result of local histogram equalization applied to (a), using a neighborhood of size 3 × 3.

Filtering

- Filtering in signal processing is the process of accepting (passing) or rejecting certain frequency components
- Lowpass filters preserve low frequencies, rejecting the high ones
- Highpass filters preserve high frequencies, rejecting the low ones

Filtering

- Lowpass filters are used in image processing for removal or reduction of noise

- Highpass filters are used in image processing for edge detection and distinguishing of small details

Filtering

- Noise in an image is resulted in corruption of image intensities. Noise may corrupt all pixels in an image (additive or multiplicative noise) or some of them (impulse noise)
- A natural way to suppress or remove noise in the spatial domain is to perform some kind of averaging of intensities affected by noise.
- This averaging can be "literal" (by replacement of noisy intensities with arithmetic mean over their local neighborhood), linear (by finding a weighted mean over their local neighborhood, and nonlinear (by some kind of nonlinear averaging over their local neighborhood

Models of Noise

- Models of noise
- Additive noise $$g(x,y)=f(x,y)+\eta(x,y)$$
- Multiplicative (speckle) noise
$$g(x,y)=f(x,y)\times\eta(x,y)$$
- Impulse noise
$$g(x,y)=\begin{cases} p_n, \eta(x,y) \\ 1-p_n, f(x,y) \end{cases}$$

where p_n is the probability of distortion (p_n in percents is called the corruption rate) $p_n \cdot 100\%$

The Goal of Noise Filtering

- The goal of noise filtering is to obtain the best approximation $\hat{f}(x,y)$ of the "ideal" image $f(x,y)$ in terms of the mean square error (root mean square error) or, which is the same, in terms of PSNR
- Filtering is utilized through a filtering operator applied to a noisy image:

$$\hat{f}(x,y) = F\big(g(x,y)\big)$$

"Ideal" Filtering

- If there are k realizations of the same image scene corrupted by the same kind of additive noise, and $k \to \infty$, then the original image can be restored by taking component-wise mean of its realizations: $g_i(x,y) = f(x,y) + \eta_i(x,y); i = 1, 2, \ldots, k, \ldots$

$$\hat{f}(x,y) = \frac{1}{k}\sum_{i=1}^{k} g_i(x,y)$$

- The larger is k, the closer $\hat{f}(x,y)$ is to $f(x,y)$

Linear Filters

- Linear filtering is utilized through a linear operator
- The operator F is called linear, if for arbitrary constants a and b the following property holds:

$$F\left(af\left(x,y\right)+b\eta(x,y)\right)=F\left(af\left(x,y\right)\right)+F\left(b\eta(x,y)\right)=$$
$$=aF\left(f\left(x,y\right)\right)+bF\left(\eta(x,y)\right)$$

Linear Filters

- A filter, which is represented by a linear operator is called a linear filter
- Ideally, a linear filter may separate an image from additive noise. If F is a linear filter, then

$$F\left(g\left(x,y\right)\right)=F\left(f\left(x,y\right)+\eta(x,y)\right)=F\left(f\left(x,y\right)\right)+F\left(\eta(x,y)\right)$$

and an image can be restored as follows

$$f\left(x,y\right)=F^{-1}\left(F\left(f\left(x,y\right)\right)\right)=F^{-1}\left(F\left(g\left(x,y\right)\right)-F\left(\eta(x,y)\right)\right)$$

Linear and Nonlinear Filters

- As $F\left(\eta(x,y)\right)$ can be estimated only in the frequency domain and this estimation is possible only for some particular kinds of noise, the "ideal" method has a limited applicability

- Speckle and impulse noise cannot be separated from an image using a linear filter. Thus, for their filtering nonlinear filters shall be used

Linear and Nonlinear Filters

- A filter, which is represented by an operator, which is not linear, is called a nonlinear filter
- Linear filters can be implemented in the frequency domain (where they originally were applied by Norbert Wiener in 1940s) and in the spatial domain
- Most of nonlinear filters can be implemented only in the spatial domain (except some specific nonlinear transformations of spectra that are applicable only in the frequency domain)

Filter Design

- When any new filter is designed, then to test its efficiency, the following model should be used:
- ➢ A clear image should be artificially corrupted using a certain kind of noise
- ➢ Then an artificially corrupted image should be filtered, and the result shall be compared to the “ideal” clear image in terms of PSNR
- ➢ A filter showing stable good results, being applied to different images (when compared to other filters), should be considered good

Modeling of Additive Noise

- To add noise to a clear image, it is necessary to generate an image-noise with a desired distribution (Gaussian is most frequently used) using a random numbers generator, with exactly the same sizes as the ones of the clear image, the same mean as the one of the clean image and σ_n (standard deviation) equal to $0.1\sigma - 0.5\sigma$ of the clear image.
- Then this noise shall be added to the clear image component-wise and their common mean shall be component-wise subtracted from the sum

$$g(x,y) = f(x,y) + \eta(x,y) - mean$$

Modeling of Additive Noise

- This procedure for generation of additive Gaussian noise is equivalent to generation of a 2-D field of random numbers with Gaussian (normal) distribution, with zero mean and standard deviation σ_n equal to $0.1\sigma - 0.5\sigma$ of the clear image.
- Then this noise shall be added to the clear image component-wise

$$g(x,y) = f(x,y) + \eta(x,y)$$

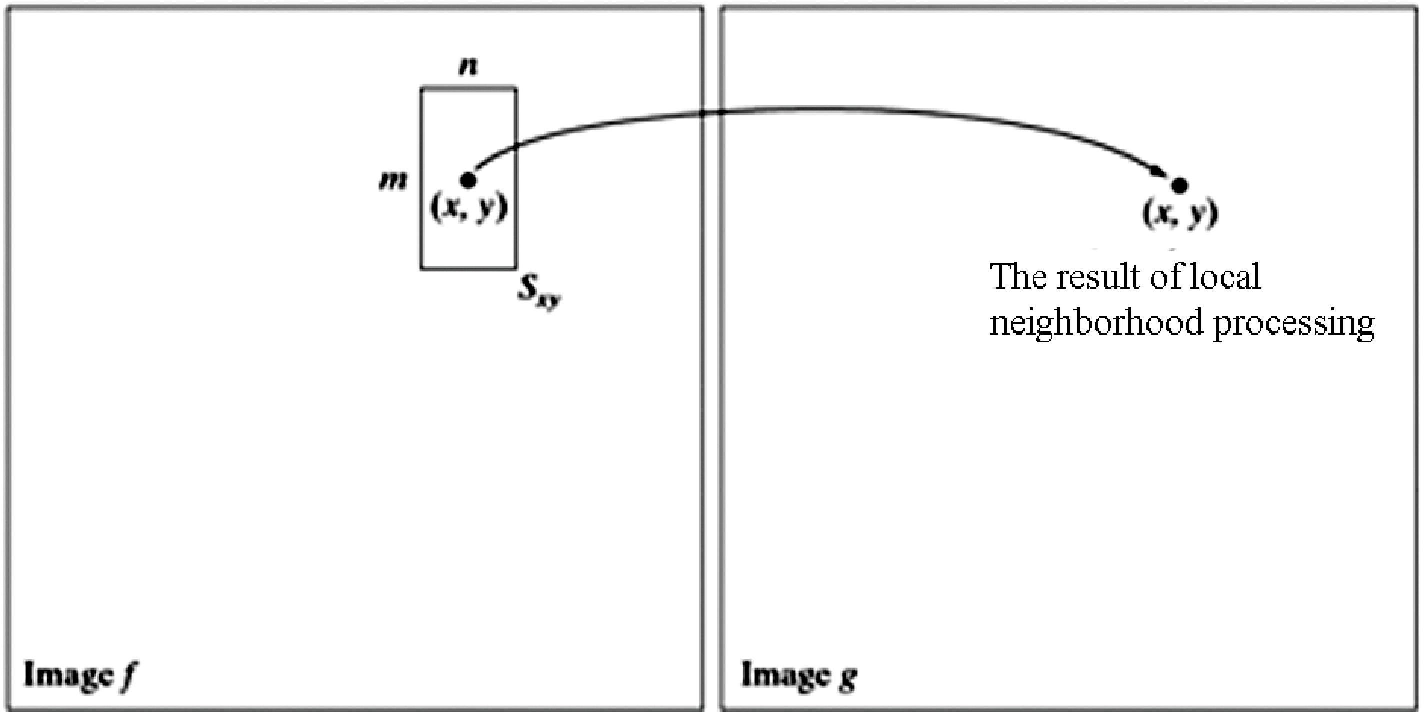
Spatial Domain Filtering

- Spatial domain filters are reduced to processing of a local neighborhood of every pixel
- Filtering creates a new pixel with coordinates equal to the coordinates of the center of the neighborhood and whose value is the result of the filtering operator applied to the neighborhood of the processed pixel
- Filtering of different pixels is typically independent and in such a case <u>can be done in parallel</u>

Spatial Domain Filtering

- Any lowpass spatial domain filter averages intensities (linearly or non-linearly) over an
  n x m local neighborhood of each pixel and smooths an image due to this averaging.
- This means that image details whose size is about 0.5n x 0.5m may become indistinguishable. This directly follows from the Nyquist Theorem
- Lowpass spatial domain filters are used for noise reduction. Averaging over each n x m local neighborhood , they “dissolve” a noise in an image

Spatial Domain Filtering



Linear Spatial Domain Filtering

- Linear spatial filtering of an image of size M x N with a filter of size m x n is defined by the expression

$$\hat{f}(x,y)=\sum_{s=-a}^{a}\sum_{t=-b}^{b}w(s,t)\,g(x+s,y+t)$$

where $g(x,y)$ is an image to be processed, $a=(m-1)/2$, $b=(n-1)/2$, and the weights $w(s,t)$ create a filter kernel

- The operation determined by this formula is called a linear convolution with a respective kernel W

Convolution

- Convolution of two continuous functions $f(t)$ and $w(t)$ is a process of flipping (rotating by 180^0) one function about its origin and sliding it past the other

$$g(t) = f(t) * w(t) = \int_{-\infty}^{\infty} w(\tau) f(t \div \tau) d\tau$$

where $\div$ stands for the flipping, τ is the displacement needed to slide one function past the other

Discrete Convolution

- Convolution of two discrete functions $f(t)$ and $w(t)$ is also a process of flipping (rotating by 180^0) one function about its origin and sliding it past the other

$$g(t) = f(t) * w(t) = \sum_{k=0}^{N-1} w(k) f(t \div k); t = 0, 1, \ldots, N-1$$

where $\div$ stands for the flipping (shift corresponding to the basis, where the convolution is taken), k is the displacement needed to slide one function past the other

Linear Spatial Domain Filtering

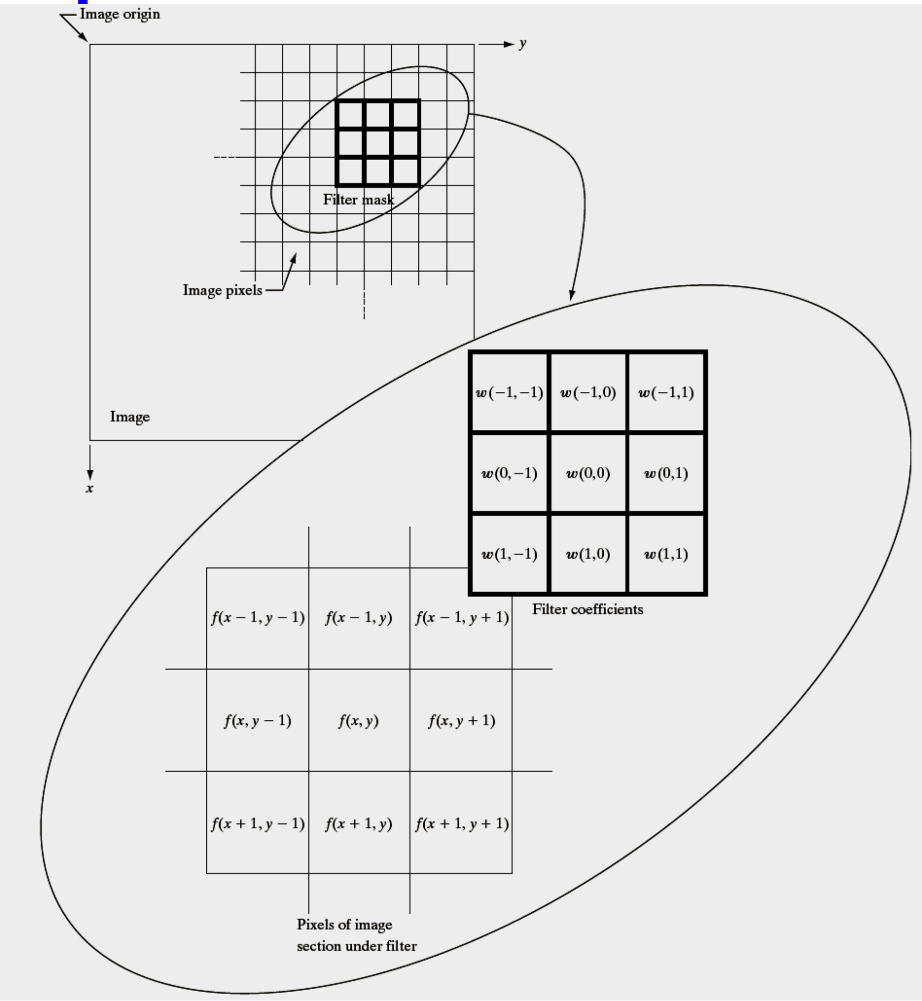


FIGURE 3.28 The mechanics of linear spatial filtering using a 3 × 3 filter mask. The form chosen to denote the coordinates of the filter mask coefficients simplifies writing expressions for linear filtering.

Boundary Effects in Spatial Domain Filtering

- To process image boundaries, it is necessary to extend an image in all directions, otherwise it will not be possible to build local neighborhoods for the border pixels

- The simplest way of such an extension is zero-padding . However, this method always creates a “black” frame along the image boundaries.

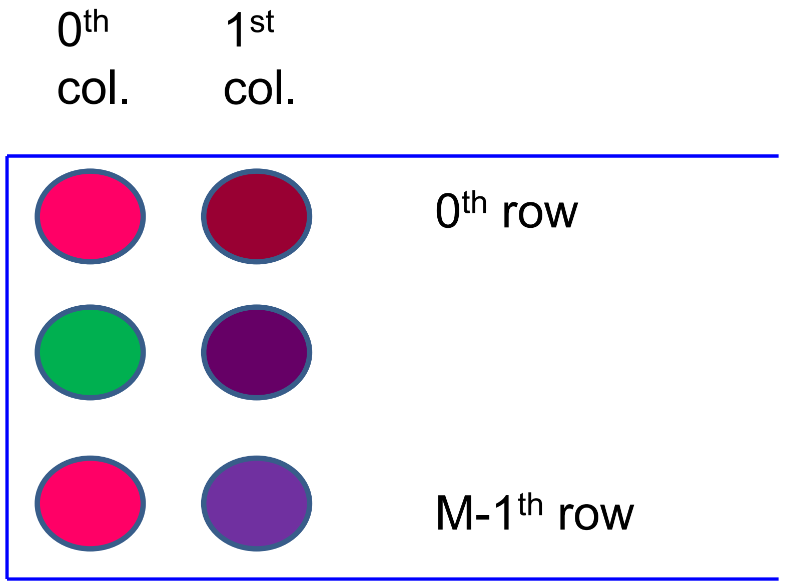
Boundary Effects in Spatial Domain Filtering

- To take care of boundary effects, cropping (mirroring) shall be used. An *N* x *M* image, before it is processed by a spatial domain filter with an *n* x *m* kernel, shall be extended to an $N+\left\lfloor \frac{n}{2} \right\rfloor \times M+\left\lfloor \frac{m}{2} \right\rfloor$ image as follows

$$\tilde{f}(x,y)=\begin{cases} f(x,y); x=0,1,\ldots,N-1, y=0,1,\ldots,M-1 \\ f(-x,y); x=-1,\ldots,-\lfloor n/2 \rfloor, y=0,1,\ldots,M-1 \\ f((N-1)-(x-N+1),y); x=N,\ldots,N+\lfloor n/2 \rfloor-1, y=0,1,\ldots,M-1 \\ f(x,-y); x=0,1,\ldots,N-1, y=-1,\ldots,-\lfloor m/2 \rfloor \\ f(x,(N-1)-(y-N+1)); x=0,1,\ldots,N-1, y=N,\ldots,N+\lfloor m/2 \rfloor-1 \\ f(-x,-y); x,y=-1,-2,\ldots \\ f((N-1)-(x-N+1),(N-1)-(y-N+1)); x,y=N,N+1,\ldots \end{cases}$$

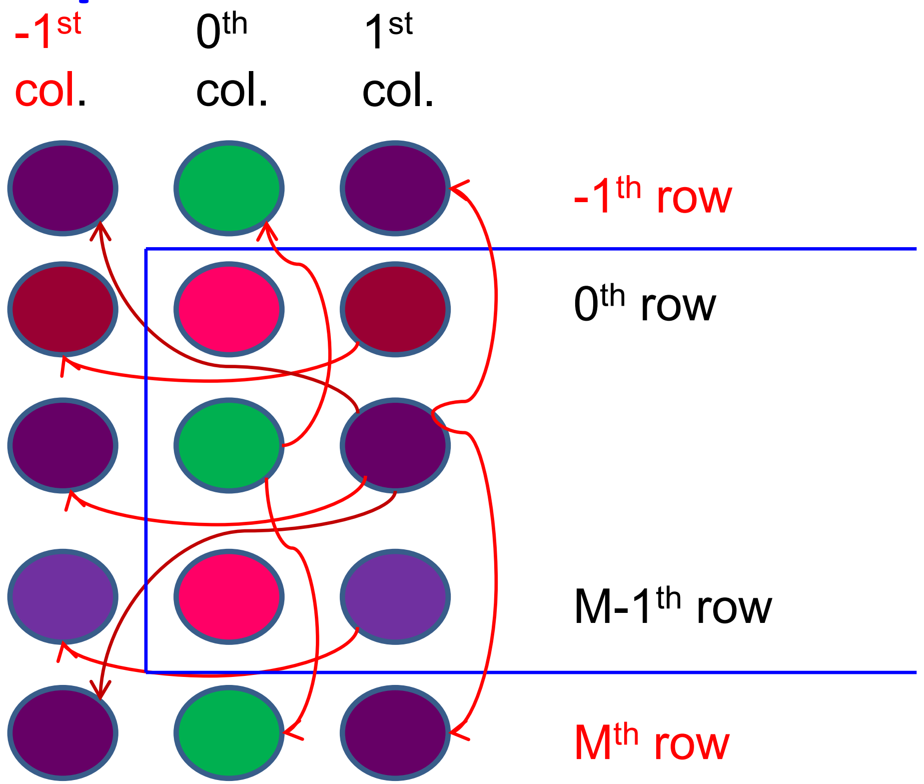
Boundary Effects in Spatial Domain Filtering

Let us consider for example, a top left corner of an image



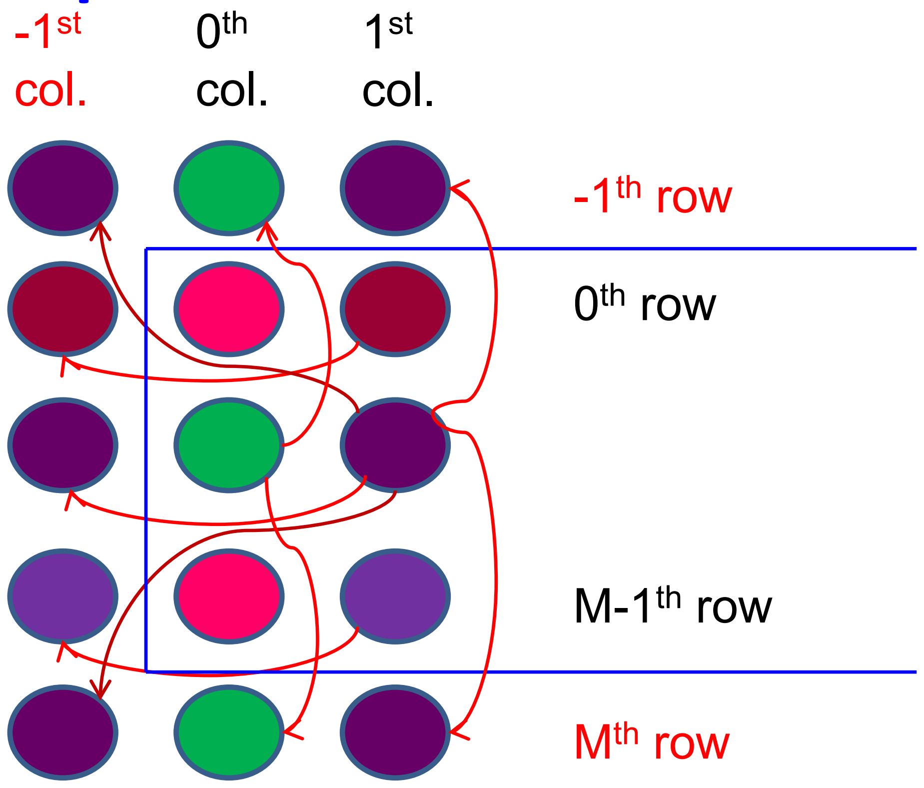
Suppose we need to apply a spatial domain filter with a 3x3 window.
This means that the following extension shall be made

Boundary Effects in Spatial Domain Filtering



Mirroring for a 3x3 local neighborhood window

Boundary Effects in Spatial Domain Filtering



Mirroring for a 3x3 local neighborhood window

Arithmetic Mean Filter – the Simplest Spatial Domain Linear Filter

- Arithmetic mean filter replaces the intensity value in each pixel by the local arithmetic mean taken over a local *n* x *m* processing window:

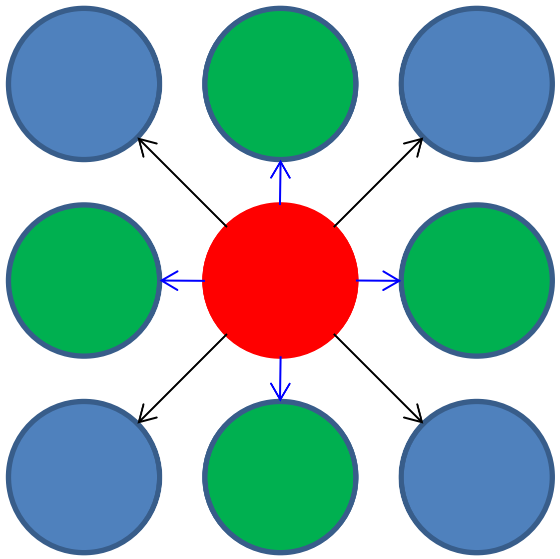
$$\hat{f}(x,y)=\frac{1}{nm}\sum_{s=-a}^{a}\sum_{t=-b}^{b}g(x+s,y+t)$$

- This corresponds to the filter kernel (mask)

$$\begin{pmatrix} \frac{1}{nm} & \frac{1}{nm} & \cdots & \frac{1}{nm} \\ \frac{1}{nm} & \frac{1}{nm} & \cdots & \frac{1}{nm} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{1}{nm} & \frac{1}{nm} & \cdots & \frac{1}{nm} \end{pmatrix}$$

Smart (Gaussian-like) Filter Kernel

- Vertical and horizontal adjacent pixels (the green ones in the picture) in a 3x3 window are located “closer” to the center of the window than the diagonal pixels. Evidently, the distance from a center of a square to any of its sides is shorter than the one to its vertices:



Smart (Gaussian-like) Filter Kernel

- So it can be reasonable to emphasize contribution of the central pixel and its closest neighbors to the resulting intensity value. This can be done by the following filter mask (kernel)

$$\frac{1}{16}\begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{pmatrix}$$